Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A method, comprising:

in an object oriented run-time environment:

a) invoking a second method from a first method, said invoking comprising providing an identification of said first method and a class that said first method is a part of;

b) identifying a plug-in module for said first method based upon said identification, said plug-in module containing a handler method;

c) executing said handler method to report and/or record information about ~~perform a output function for~~ said first method; and,

d) executing said first method from a point beyond where said second method was invoked.

2. (Original) The method of claim 1 wherein said executing of said handler method causes an entry time for said first method to be recorded .

3. (Original) The method of claim 1 wherein said executing of said handler method causes an exit time for said first method to be recorded.

4. (Original) The method of claim 1 wherein said executing of said handler method causes a counter maintained for said first method to be incremented.

5. (Original) The method of claim 1 wherein said executing of said handler method causes an input parameter value of said first method to be recorded.

6. (Original) The method of claim 1 wherein said executing of said handler method causes a returned value of said first method to be recorded.

7. (Original) The method of claim 1 wherein said first method is a constructor.

8. (Original) The method of claim 1 further comprising creating, prior to said invoking, an object having an input parameter value of said first method.

9. (Original) The method of claim 1 wherein said invoking further comprises providing an input parameter value of said first method.

10. (Original) The method of claim 1 wherein said invoking further comprises identifying where said first method's instructions can be found in memory.

11. (Currently Amended) The method of claim 1 further comprising, after said executing said first method from a point beyond where said second method was invoked ~~a portion of said first method~~:

e) invoking a third method from said first method because said first method is about to reach an exit point, said second method having been invoked from said first method because an entry point of said first method had just been reached;

f) re-identifying said plug-in module for said first method as a consequence of said invoking a third method;

g) re-executing said handler method to report and/or record information about ~~perform a output function for~~ said first method; and,

h) executing a remaining portion of said first method through said exit point.

12. (Currently Amended) The method of claim 1 further comprising, after said executing said first method from a point beyond where said second method was invoked ~~a portion of said first method~~:

e) flowing from said first method to a third method

f) invoking said second method from said third method, said invoking comprising providing an identification of said third method and a second class that said third method is a part of;

g) identifying said plug-in module for said third method based upon said third method and second class identification;

h) executing said handler method to report and/or record information about ~~perform a output function for~~ said third method; and,

i) executing a portion of said third method from a point beyond where said second method was invoked.

13. (Currently Amended) The method of claim 12 wherein g) further comprises ~~comprising~~ also identifying a second plug-in module for said third method based upon said third method and second class identification, said second plug-in module containing a second handler method.

14. (Currently Amended) The method of class 13 further comprising also executing said second handler method to ~~perform a~~ report and/or record different information about said third method than what said first handler method reported and/or recorded about said third method~~output function than said output function for said third method~~.

15. (Currently Amended) The method of claim 14 wherein a first object is called to execute said first method and a second ~~method~~ object is called to execute said third method~~object~~.

16. (Currently Amended) The method of claim 15 wherein said object oriented run-time environment is a Java object oriented environment.

17. (Currently Amended) The method of claim 1 wherein said invoking further comprises providing said first method's signature, said first method's signature comprising:

said identification of said first method;

said identification of said class that said first method is a part of; and,

said first method's arguments.

18. (Currently Amended) One or more machine readable media containing instructions which when executed by one or more computing systems cause a method to be performed, said method, comprising:

in an object oriented run-time environment:

a) invoking a second method from a first method, said invoking comprising providing an identification of said first method and a class that said first method is a part of;

b) identifying a plug-in module for said first method based upon said identification, said plug-in module containing a handler method;

c) executing said handler method to report and/or record information about ~~perform a output function for~~ said first method; and,

d) executing said first method from a point beyond where said second method was invoked.

19. (Original) The one or more machine readable media of claim 18 wherein said executing of said handler method causes an entry time for said first method to be recorded .

20. (Original) The one or more machine readable media of claim 18 wherein said executing of said handler method causes an exit time for said first method to be recorded.

21. (Original) The one or more machine readable media of claim 18 wherein said executing of said handler method causes a counter maintained for said first method to be incremented.

22. (Original) The one or more machine readable media of claim 18 wherein said executing of said handler method causes an input parameter value of said first method to be recorded.

23. (Original) The one or more machine readable media of claim 18 wherein said executing of said handler method causes a returned value of said first method to be recorded.

24. (Original) The one or more machine readable media of claim 18 wherein said first method is a constructor.

25. (Original) The one or more machine readable media of claim 18 further comprising creating, prior to said invoking, an object having an input parameter value of said first method.

26. (Original) The one or more machine readable media of claim 18 wherein said invoking further comprises providing an input parameter value of said first method.

27. (Original) The one or more machine readable media of claim 18 wherein said invoking further comprises identifying where said first method's instructions can be found in memory.

28. (Currently Amended) The one or more machine readable media of claim 18 further comprising, after said executing said first method from a point beyond where said second method was invoked ~~a portion of said first method~~:

e) invoking a third method from said first method because said first method is about to reach an exit point, said second method having been invoked from said first method because an entry point of said first method had just been reached;

f) re-identifying said plug-in module for said first method as a consequence of said invoking a third method;

g) re-executing said handler method to report and/or record information about ~~perform a output function for~~ said first method; and,

h) executing a remaining portion of said first method through said exit point.

29. (Currently Amended) The one or more machine readable media of claim 18 further comprising, after said executing said first method from a point beyond where said second method was invoked ~~a portion of said first method~~:

e) flowing from said first method to a third method

f) invoking said second method from said third method, said invoking comprising providing an identification of said third method and a second class that said third method is a part of;

g) identifying said plug-in module for said third method based upon said third method and second class identification;

h) executing said handler method to report and/or record information about ~~perform a output function for~~ said third method; and,

i) executing a portion of said third method from a point beyond where said second method was invoked.

30. (Currently Amended) The one or more machine readable media of claim 29 wherein g) further comprises ~~comprising~~ also identifying a second plug-in module for said third method based upon said third method and second class identification, said second plug-in module containing a second handler method.

31. (Currently Amended) The one or more machine readable media of class 30 further comprising also executing said second handler method ~~perform a~~ report and/or record different information about said third method than what said first handler method reported and/or recorded about said third method~~output function than said output function for said third method~~.

32. (Currently Amended) The one or more machine readable media of claim 31 wherein a first object is called to execute said first method and a second object ~~method~~ is called to execute said third method~~object~~.

33. (Currently Amended) The one or more machine readable media of claim 32 wherein said object oriented run-time environment is a Java object oriented environment.

34. (Currently Amended) The one or more machine readable media of claim 18 wherein said invoking further comprises providing said first method's signature, said first method's signature comprising:

said identification of said first method;

said identification of said class that said first method is a part of; and,

said first method's arguments.

COMMENTS

The enclosed is responsive to the Examiner's Office Action mailed on August 23, 2006. At the time the Examiner mailed the Office Action claims 1-34 were pending. By way of the present response the Applicant has: 1) amended claims 1, 11-18 and 28-34; 2) has not added any claims; 3) has not canceled any claims. As such, claims 1-34 remain pending. The Applicant respectfully requests reconsideration of the present application and the allowance of claims 1-34.

Claim Objections

The Examiner has objected to claims 1 and 18 for their recital of the term "a output". That term has presently been stricken thereby removing the basis for the Examiner's rejection.

Claim Rejections Under 35 U.S.C. § 112

Claims 11 and 28 stand rejected because of their recital of the terms "re-identifying said plug-in module" and "re-executing said handler method". According to the Examiner "it is unclear how and why the plug-in module of Claims 1 and 18 is re-identified and how and why the handler method of Claims 1 and 18 is re-executed . . . [a]pplicant's specification teaches identifying a plug-in module and executing a handler as a consequence of invoking another method." See, Examiner's Office Action, mailed 8/23/06, pg. 2.

The Applicant invites the Examiner to focus upon Figure 4b of the Applicant's specification. Figure 4b of the Applicant's specification reveals a run-time process flow (with a dotted line) which shows, for example, entry into "method_1" 405, where specially inserted bytecode instructions at entry point 409 invoke dispatch unit 430. The dispatch unit 430 identifies the appropriate plug-in module for method_1 405 (specifically, plug-in module A 460) and assists re-direction of the process flow to plug-in module A 460. Because plug-in module A 460 contains "tracing" handler 461, re-direction of the process flow to plug-in module A 460 causes the instructions of tracing handler 461 to be executed. The process flow returns to method_1 405 to execute the bytecode instructions that follow the specially inserted bytecode at entry point 409. Eventually the process flow is destined to exit method 1 causing specially inserted bytecode at the exit point 415 to again invoke dispatch unit 430. The dispatch unit 430 re-identifies plug-in module A 460 as the appropriate plug-in module. Thus the process flow, as indicate by the dotted line, is again re-directed to plug-in A module A 460, which , in turn, causes the instructions of tracing handler 461 to be re-executed.

Thus the Applicant's specification discloses re-identification of a plug-in module and re-execution of a handler method. The Examiner can view the above described embodiment as being covered by claim 11 with the following perspective: 1) "the first method" corresponds to method_1 405; 2) "the second method" corresponds to a ".entry" method call made to the dispatch unit 430 from entry point 409; 3) "the third method" corresponds to a ".exit" method call made to the dispatch unit 430 from exit point 415. For further information the Examiner

is invited to review: 1) paragraph [0066] (summarizing that the same plug-in module and tracing handler is invoked for both entry to and exit from method_1 405); and, 2) paragraphs [0081] through [0085] (discussing ".entry" and ".exit" method calls to the dispatcher).

Claims 12 and 29 stand rejected because of their recital of the terms "identifying said plug-in module" and "executing said handler method". According to the Examiner, ". . . it is unclear how and why the handler method of Claims 1 and 18 is executed. See, Examiner's Office Action mailed 8/23/06, p. 2.

The Examiner is again invited to refer to Figure 4b of the Applicant's specification, and moreover, to continue following the process flow after it exits method_1 405 at exit point 415 and enters method_2 406 at entry point 410. From the process flow observed in Figure 4b it is clear that method_2 406 follows the same process as method_1 405. That is, for both entry point 410 and exit point 417, plug-in module A identifies dispatch unit 430 and the process flow flows through handler 461. Therefore the same plug-in module and handler are identified and executed for two different methods (method_1 405 and method2 406). The Examiner can view the embodiment discussed above as being covered by claim 12 with the following perspective: 1) "the first method" corresponds to method_1 405; 2) "the second method" corresponds to a ".entry" method call made to the dispatch unit 430 (once at element a) and again at element f)); and 3) "the third method" corresponds to method_2 406.

Claims 11-12 and 28-29 stand rejected for the claim element "a portion of said first method." The Applicant has stricken this language thereby removing the basis for the Examiner's rejection.

Claims 15 and 32 stand rejected for their recitation of the claim element "execute third object". The Applicant has amended claims 15 and 32 to recite ". . . wherein a first object is called to execute said first method and a second object s called to execute said third method. The Applicant respectfully submits that the basis for the Examiner's rejection has been removed by way of the above amendment.

Claims 17 and 34 stand rejected for their recitation of the claim element "said identification of said class". The Applicant respectfully submits that claims 17 and 34 meet the requirements of 35 U.S.C. § 112, ¶ 2 as written. Specifically, element a) of base claims 1 and 18 recite "identification of said first method and a class". That is, both a method and a class are identified. Therefore, the meaning of the claim element "said identification of said class" is sufficiently clear and does require further amendment.

Claim Rejections Under 35 U.S.C. § 101

The Examiner has rejected all claims as failing to be directed to statutory subject matter. The Applicant disagrees with the Examiner's initial rejection of

the claims under 35 U.S.C. § 101. However, in order to improve the clarity of element c) in independent claims 1 and 18, the Applicant has amended element c) of independent claims 1 and 18 to recite "executing [the] handler method to report and/or record information about [the] first method". As a side-effect, the Applicant submits that the above described amendment causes the claim to be acceptable to the Examiner's definition of the requirements of 35 U.S.C. § 101. Specifically, a result in the form of a report and/or record of information about the first method is specifically recited. The Applicant therefore respectfully submits that the rejections under 35 U.S.C. § 101 should be removed.

Claim Rejections Under 35 U.S.C. § 102(e)

Independent claims 1 and 18 stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Pub'd App. No. 2005/0039171, hereinafter "Avakian et. al.". The Applicant commends the Examiner for the finding of the pertinent Avakian et. al. reference. However, the Applicant respectfully submits that Avakian et. al. reference. Recall from the Applicant's discussions above regarding the rejections of claims 11, 12 and 28, 29 under 35 U.S.C. § 112, ¶ 2 that the Applicant's disclosure articulates method calls made to a dispatcher at method entry points and method exit points. The Examiner was advised to review paragraphs [0081] through [0085] of the Applicant's specification concerning the ".entry" and ".exit" methods. These paragraphs, and Figure 6A of the Applicant's specification to which paragraphs [0081] and [0085] refer, clearly articulate that the parameters of the .entry and .exit methods include an

identification of the calling method and its class. See, e.g., Applicant's specification, Figure 6A, showing "classid" and "methodid" for each of method calls 643a, 644a, 645a, 646a. From this view, moreover, element a) of independent claims 1 and 18 recites (emphasis added):

> invoking a second method from a first method, said invoking comprising providing an identification of said first method and a class that said first method is a part of

The Applicant respectfully requests that Avakian et. al. fails at least to disclose this element. That is, the function calls made by the specially inserted bytecode of Avakian do not include an identification of the calling method nor its class. Figure 7 of Avakian et. al. discloses a number of functions calls made by the specially inserted bytecode instructions 702, 706, 708, 716 none of which include as their parameters the identification of the instrumented method ("buy") nor its class. As such, Avakian fails to anticipate independent claims 1 and 18. Therefore independent claims 1 and 18 are allowable over the Avakian et. al. reference.

Because the Applicant has demonstrated the patentability of all pending independent claims, the Applicant respectfully submits that all pending claims are allowable. The Applicant's silence with respect to the dependent claims should not be construed as an admission by the Applicant that the Applicant is complicit with the Examiner's rejection of these claims. Because the Applicant has demonstrated the patentability of the independent claims, the Applicant need not substantively address the theories of rejection applied to the dependent claims.

CONCLUSION

For the reasons provided above, the Applicant respectfully submits that the current set of claims is allowable. If the Examiner believes an additional telephone conference would expedite or assist in the allowance of the present application, the Examiner is invited to call Robert B. O'Rourke at (408) 720-8300.

Authorization is hereby given to charge our Deposit Account No. 02-2666 for any charges that may be due.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 11/2, 2006

Robert B. O'Rourke
Reg. No. 46,972

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, CA 90025-1030
(408) 720-8300